Renesas Synergy™ Platform

NetX Duo™ NAT Module Guide

Introduction

This module guide enables you to effectively use a module in your own design. When you complete this guide, you will be able to add this module to your own design, configure it correctly for the target application, and write code using the included application project code as a reference and an efficient starting point. References to API descriptions and suggestions to other application projects that demonstrate advanced uses of the module are included and should be valuable in creating more complex designs.

The IP Network Address Translation (NAT) solves the problem of a limited number of Internet IPv4 addresses that arises when multiple devices need access to the Internet, but only one IPv4 Internet address is assigned by the Internet Service Provider (ISP). A NAT-enabled router is installed between the public and private network to translate between internal private IPv4 addresses and assigned public IPv4 address, so devices on the private network can share the same public IPv4 address.

This module guide presents key elements related to NetX Duo NAT Module implementation on the Renesas Synergy Platform, especially the addition and configuration of the NetX Duo NAT to a project. For details on the module's operation, see the document, *NetX ™ Duo NAT User Guide for the Renesas Synergy™ Platform*, included in the *X-Ware™ and NetX™ Component Documents for Renesas Synergy™* zip file available from the Renesas Synergy Gallery (www.renesas.com/synergy/ssp).

Note: NAT is not available for NetX; the NAT protocol does not support IPv6 packet forwarding.

Contents

1. NetX Duo NAT Module Features

- NetX NAT supports the following RFCs:
  - RFC 2663: IP Network Address Translator (NAT) Terminology and Considerations
  - RFC 3022: Traditional IP Network Address Translator (Traditional NAT)
  - RFC 4787: Network Address Translation (NAT) Behavioral Requirements for Unicast User Datagram Protocol (UDP)
- NetX NAT provides the following high-level APIs:
  - Creating and deleting a NAT server
  - Enabling and disabling NAT in NetX Duo
  - Set callbacks for NAT to notify application if NAT entry table is full
  - Creating static inbound entries (rules) in the NAT table

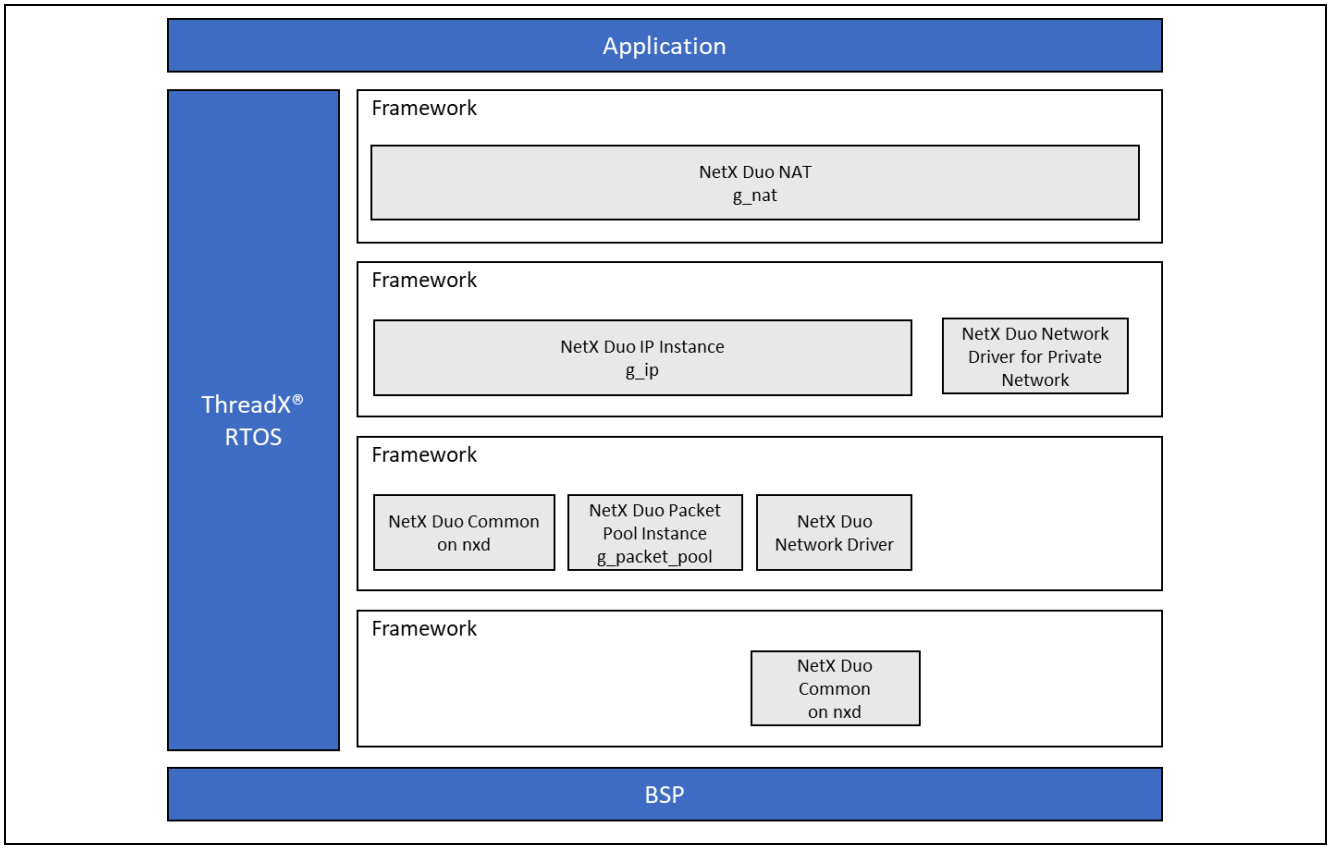


Figure 1. NetX Duo NAT Module Block

Note: In the figure above, the NetX Duo Network Driver modules has multiple implementation options available. See the description just after the module stack figure in section 4 for additional details.

2. NetX Duo NAT Module APIs Overview

The NetX Duo NAT Module defines APIs for creating, deleting and enabling operations. The following table includes a complete list of the available APIs, an example API call, and a short description of each API. A table of status return values follows the API summary table.

Table 1. NetX Duo NAT Module API Summary

Function Name	Example API Call and Description
nx_nat_create	`nx_nat_create(nat_ptr, ip_ptr, global_interface_index, nat_cache, NX_NAT_ENTRY_CACHE_SIZE);` Create a NAT Instance in which the network interface is the global interface (not the local/private network) with the specified network index.

Function Name	Example API Call and Description
nx_nat_delete	`nx_nat_delete (nat_ptr);` Delete a NAT instance.
nx_nat_enable	`nx_nat_enable (nat_ptr);` Enable the NAT router.
nx_nat_disable	`nx_nat_disable (nat_ptr);` Disable the NAT router.
nx_nat_cache_notify_set	`nx_nat_cache_notify_set(nat_ptr, cache_full_notify_cb);` Set the NAT cache full notify function to a user-defined notify function.
nx_nat_inbound_entry_create	`nx_nat_inbound_entry_create(nat_ptr, entry_ptr, IP_ADDRESS(192,168,2,2), 5001, 5001, NX_PROTOCOL_TCP);` Create an inbound translation table entry. This is typically used by application servers to allow external host to initiate a connection with an internal host.
nx_nat_inbound_entry_delete	`nx_nat_inbound_entry_delete(nat_ptr, delete_entry_ptr);` Delete an inbound translation table entry.

Note: For details on operation and definitions for the function data structures, typedefs, defines, API data, API structures, and function variables, review the associated Express Logic User's Manual listed in the Reference section.

Table 2. Status Return Values

Name	Description
NX_SUCCESS	Successful NAT function
NX_PTR_ERROR*	Invalid input pointer parameter
NX_CALLER_ERROR*	Invalid caller (for example, must be a thread) of a service
NX_NAT_PARAM_ERROR*	Invalid non pointer input
NX_NAT_CACHE_ERROR*	Cache memory not 4-byte aligned, or it is too small
NX_NAT_PORT_UNAVAILABLE	Invalid external port for creating static entry
NX_NAT_ENTRY_NOT_FOUND	Entry to delete is not found in Cache table
NX_NAT_ENTRY_TYPE_ERROR*	Invalid entry (not static) to delete

Note: Lower level drivers may return common error codes. See the *SSP User's Manual* API References for the associated module for a definition of all relevant status return values.
*These error codes are only returned if error checking is enabled. See NetX Duo User Guide for the Renesas Synergy™ Platform for details on error-checking services.

3. NetX Duo NAT Module Operational Overview

A NAT-enabled router typically has two network interfaces: one connected to the public Internet, the other connected to the private network. A typical router in this setup is responsible for routing IP datagrams between the private network and the public network based on the destination IP address. A NAT-enabled router performs address translation before routing an IPv4 datagram between the public and the private interface. Translation is established for each TCP or UDP session, based on the internal source address and source port number, as well as the external destination address and destination port number. For the ICMP echo request and response datagram, the Internet Control Message Protocol (ICMP) query ID is used instead of the port number.

Typically, connections across the NAT boundary are initiated by the hosts on the private network sending outbound packets to an external host. In these cases, a local host is usually assigned a dynamic (temporary) IP address. It is also possible to have connections initiated in the opposite direction if the private network has 'servers' (such as HTTP or FTP) that accept client requests from the external network. NAT applications typically assign these local hosts a static (permanent) IP address port.

The following three illustrations and accompanying steps depict the sequence of events when sending packets through a NAT router.

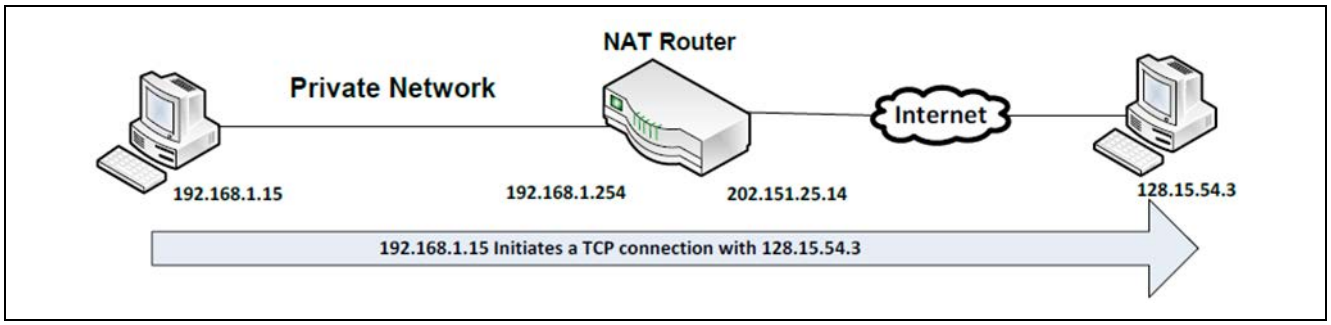


Figure 2. Connection start

1. A local Client transmits a TCP SYN message to a web server on the external network. The Client address on the private network is 192.168.1.15, port number 6732; the destination address is 128.15.54.3, port number 80.
2. The packet from the Client is received on the private network interface by the NAT router. The outbound traffic rule applies to the packet: the sender's (Client's) address is translated to the NAT router's public IP address 202.151.25.14, and sender (Client) source port number is translated to the TCP port number 2015 for transmission out on the public interface.
3. The packet is then transmitted over the Internet and ultimately reaches its destination host 128.15.54.3. On the receiving side, notice in the following figure that the packet appears to have originated from 202.151.25.14, port number 2015 when it was in fact relayed from that IP address and port.

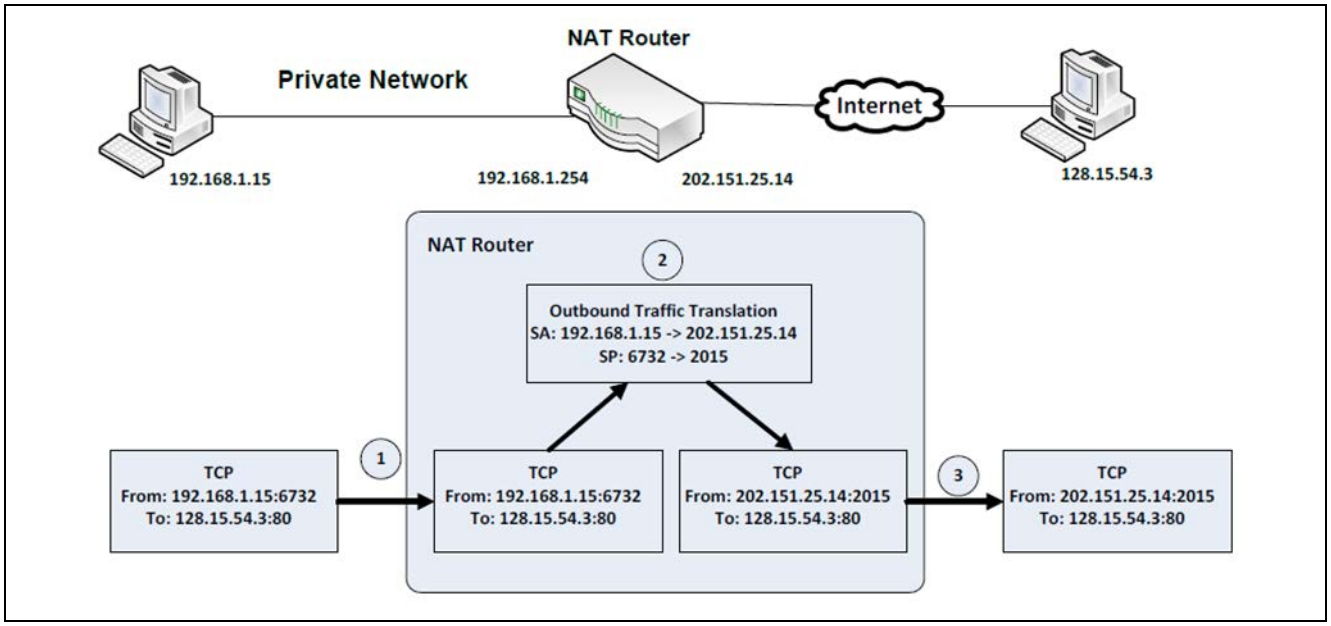


Figure 3. Network Address Translation

4. External host 128.15.54.3 sends back a response packet with the NAT router's Internet address as its destination.
5. The packet reaches the NAT router. Since this is an in-bound packet, the in-bound translation rules apply: the destination address is changed back (translated) to the original sender's (Client's) IP address: 192.168.1.15, destination port number 6732.
6. The packet is then forwarded to the Client through the interface connected to the internal network.

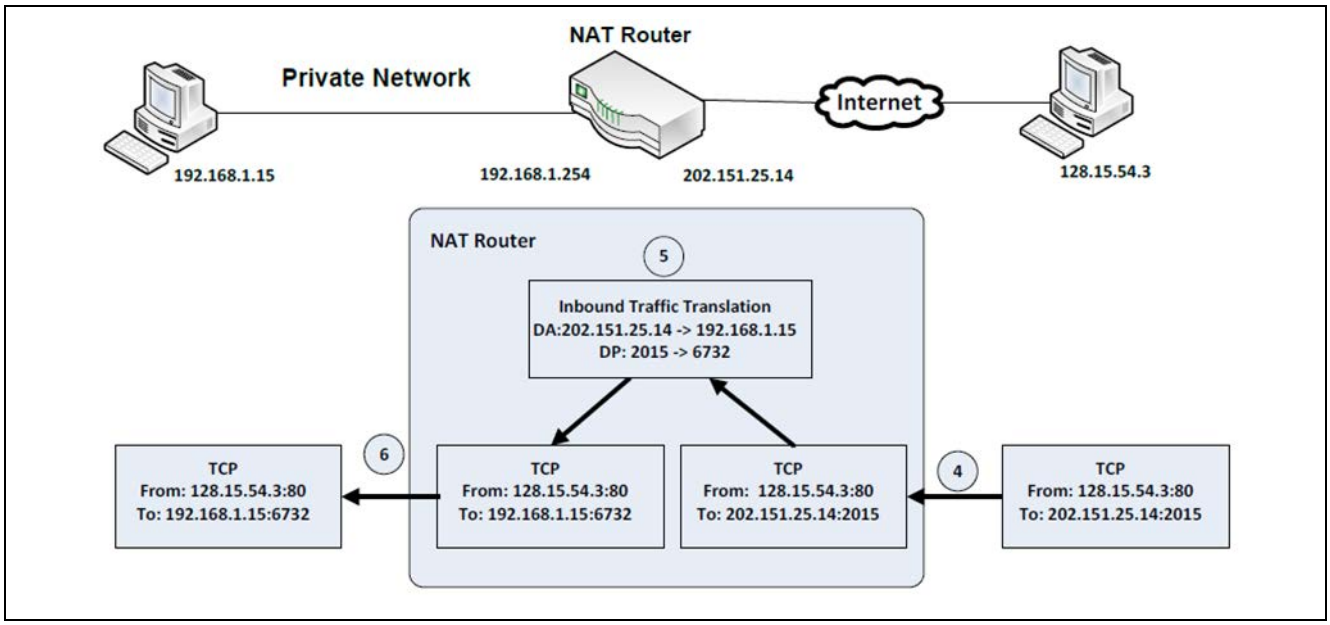


Figure 4. Packet return path

When sending packets through a NAT router, the sender's Internet network address and port number is not exposed to other hosts on the public Internet.

To keep track of the network address translations for all active connections between local and external networks, the NetX Duo NAT-enabled router maintains a translation table with information about each private host connection that includes source and destination IP address and port number.

NetX Duo NAT is intended for use on an IPv4 router. For the NAT to work, the NetX Duo must be configured to forward received packets to an internal NetX Duo NAT handler. The NAT handler determines whether the packet is received from the global network (inbound) or from the private network (outbound).

- For inbound packets, the handler determines whether it can forward (consume) the packet to the host on the private (local) network. To make the determination, the handler looks for a matching entry based on the packet destination address and port in the NAT translation table.
  - If a matching entry is found, the handler translates the destination address to the matching private host IP address and port and sends the packet to that host.
  - If it cannot find an entry, it lets the NetX Duo process the packet as normal, as if the packet was intended for the NAT device itself.
- For outbound packets, the NAT handler checks the destination IP address to determine whether it can forward the packet out onto the global network or whether the NetX Duo should handle the packet as it does normally.

  If the packet has a broadcast or loopback destination address, or an IP address that does not match the NAT device global network address, the NAT handler lets the NetX Duo handle the packet. Otherwise, the handler looks for a matching entry of the sender's IP address and port in the translation table.
  - If it finds a match, it translates the IP address and port to the local IP address and port, and forwards (consumes) the packet to the local host.
  - If a previous entry is not found, the NAT creates an entry in the NAT translation table, translates the sender's IP address for the global interface and forwards the packet to the external host.

3.1 NetX Duo NAT Module Important Operational Notes and Limitations

3.1.1 NetX Duo NAT Module Operational Notes

- To enable the NAT, add the NetX Duo Source component to the Configurator pane, and set the *NAT* property of NetX Duo Source. The prebuilt NetX Duo source library does not have NAT enabled.
- The *Maximum Physical Interfaces* property, also in NetX Duo Source component, must be changed from the default value of 1 to 2, assuming one private network and one global network. Auto-generated code creates the IP instance using the global network IP address.
- There must be two network driver instances. The project for this module uses the NetX Port ETHER framework (`sf_el_nx`) for both interfaces. However, a network interface can be on Wi-Fi, or other network media. If using two ethernet `sf_el_nx` driver instances, ensure that their names (*Name* property of the driver component) are **not** the same and that primary driver uses channel 1 and the secondary driver uses the other channel 0 (*Channel* property of the driver component).
  The MAC addresses for a dual-ported driver are in a single NetX Port Ether configuration. There is no need to change these, but you need to ensure that they are not identical between Channel 0 and Channel 1. By default, they will not be identical.
- The function specified in the *Name of the generated initialization function* property must attach the secondary interface (nx_ip_interface_attach API) and create the NAT instance (nx_nat_create API) if the *Auto initialization* property of the NAT instance is enabled (by default it is). The *Private IPv4 Address* property of the NAT instance is the local IP address of the NAT device (server).
  The *Global network interface index* property specifies the network interface that the global network uses. By default, this index is set to zero (the primary interface of the IP instance), and the secondary interface is the local network (interface index 1).
- If *Auto Initialization* is disabled, then the NAT application must attach a secondary interface and create a NAT instance before using any NAT service. After attaching the secondary interface, the application should wait for the internal NetX Duo processing to enable the link on that interface using the `nx_ip_interface_status_check` API (see the module guide project example for details on how this is done).
- At runtime, the NetX Duo NAT Framework also creates a 4-byte aligned table, or cache, to store NAT translation entries. The size of the cache is set by the *Cache Size* property of the NAT instance. The default value is 1024 bytes (a NAT translation record is 28 bytes). The minimum size of a NetX Duo NAT Translation table is three entries. This value is set in the *Minimum count for translation entry* property which defaults to three but should be set to a larger number in a busy network with many local hosts.
- By default, entries created by the NAT router when receiving inbound or outbound packets are dynamic entries. They are assigned a timeout value (*Timeout for translation_entry* property); the default value of 240 seconds is the timeout recommended by RFC 2663. When an entry timeout expires, the entry is marked for deletion. However, because the NetX Duo NAT router implementation does not have a dedicated timer task to monitor the table, NAT only checks the table for expired entries and deletes them when adding a new entry to the table. If there are no entries that have expired, and the table is full, the NetX Duo NAT router notifies the application with the cache full callback. The application can set this callback with the *nx_nat_cache_notify_set* service.
- To create static entries that never expire, the application can use the `nx_nat_inbound_entry_create` service. These entries can only be created for inbound packets and are sometimes referred to as ‘inbound rules.’ The entries are intended for server applications, to allow clients to initiate connection sessions with the server. Internally, the NAT verifies whether the global and local ports requested in the inbound rule are available. To delete these entries, the application calls the `nx_nat_inbound_entry_delete` service.
- NetX Duo NAT is configured with a range of TCP, UDP and ICMP translation ports to create unique local address: port entries for local hosts connecting with outside hosts.
  - TCP ports available for TCP translation ports are between the minimum value (*Minimum assigned port number for outbound TCP packets* property*)* and the maximum value *(Maximum assigned port number for outbound TCP packets* property*).*
  - Similarly, for UDP entries ports*, Minimum assigned port number for outbound UDP packets* and *Maximum assigned port number for outbound UDP packets* properties*.*
  - ICMP packets do not have ports; instead, the ICMP ID can be used as the entry translation port *(Minimum ICMP query identifier* and *Maximum ICMP query identifier* properties*).*

- To start NetX Duo forwarding packets using the NAT protocol, the application calls the `nx_nat_enable` service. To suspend NetX forwarding, the application calls the `nx_nat_disable` service.
- Once the NAT is enabled, the NAT thread entry application lets the internal processing for NAT handle packets transmission between the global and local networks. It can create inbound rules at any time to allow a host on the global network to reach a local host, as is typically done for servers on the local network.

3.1.2 NetX Duo NAT Module Limitations

- Internet Group Management Protocol (IGMP) is not supported. NetX Duo NAT supports only TCP, UDP, and ICMP.
- NAT protocol does not apply to IPv6 packet transmission.
- NetX Duo NAT does not include DNS or DHCP services, although NetX Duo NAT can integrate those services with its NAT operations.

See the latest SSP Release Notes for any additional module limitations.

4. Including the NetX Duo NAT Module in an Application

To include the NetX Duo NAT module in an application using the SSP configurator, use the following steps.

Note: It is assumed you are familiar with creating a project, adding threads, adding a stack to a thread and configuring a block within the stack. If you are unfamiliar with any of these items, see the *SSP User's Manual* to learn how to manage each of these important steps in creating SSP-based applications.

Add the NetX Duo NAT Module to an application by adding it to a thread using the stacks selection sequence given in the following table. (The default name for the NetX Duo NAT is `g_nat0` and this name can be changed in the associated Properties window).

Table 3. NetX Duo NAT Module Selection Sequence

Resource	ISDE Tab	Stacks Selection Sequence
g_nat0 NetX Duo NAT	Threads	New Stack> X-Ware> NetX Duo> Protocols> NetX Duo NAT

When the NetX Duo NAT module is added to the thread stack as shown in the following figure, the configurator automatically adds any needed lower-level drivers. Any drivers that need additional configuration information are box text highlighted in red. Modules with a gray band are individual, standalone modules. Modules with a blue band are shared or common and need be added only once to be used by multiple stacks. Modules with a pink band can require the selection of lower-level drivers; these are either optional or recommended as identified by block text.). If the addition of lower-level drivers is required, the module description includes "Add." Click on any pink-banded modules to bring up the "New" icon and display possible choices.

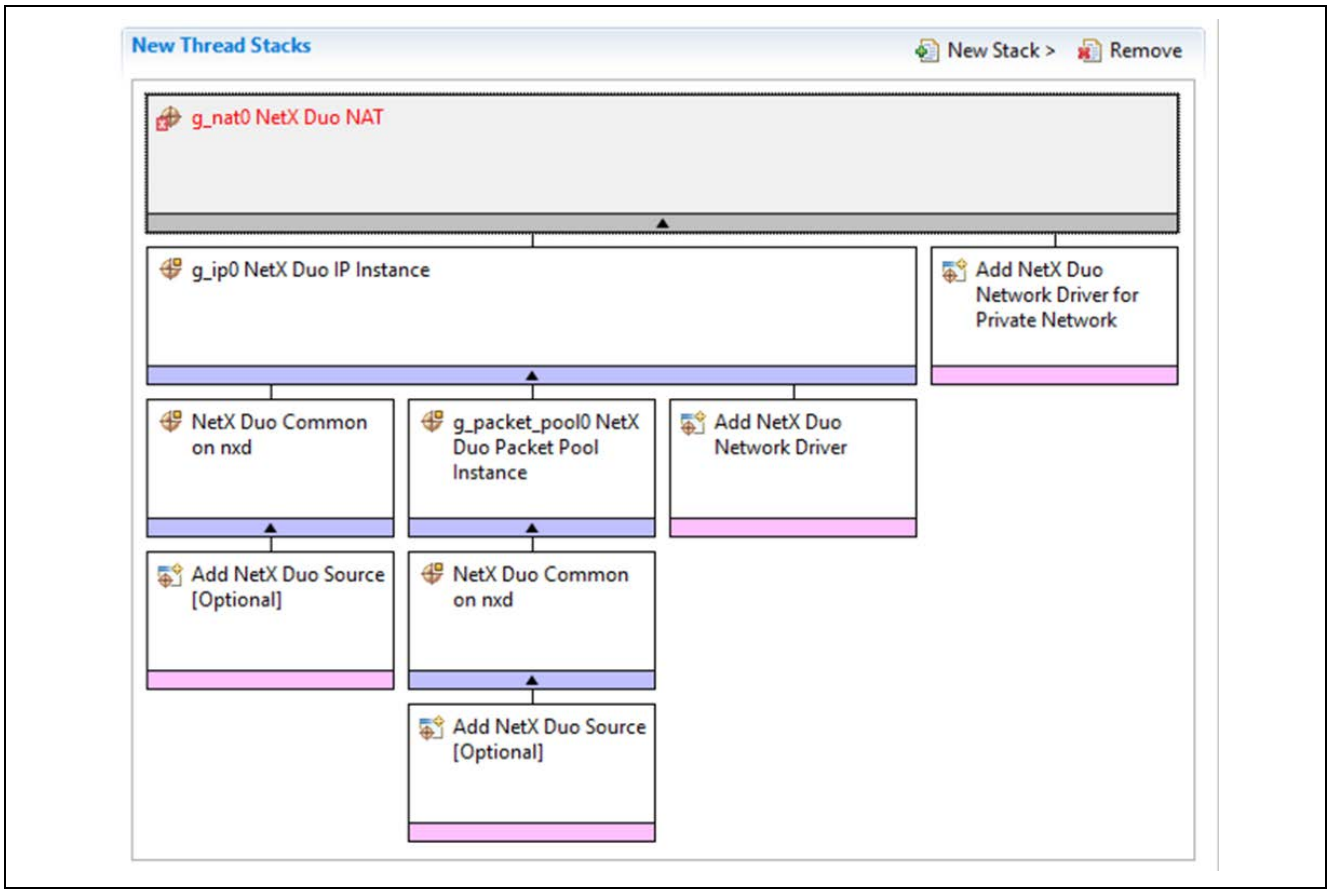


Figure 5. NetX Duo NAT Module Stack

5. Configuring the NetX Duo NAT Module

The user configures the NetX Duo NAT module for the desired operation. The SSP configuration window automatically identifies (by highlighting the block in red) the selections required for successful operation, such as interrupts or operating modes for lower-level modules. Only non-conflicting properties are available for change. Unavailable properties are 'locked' with a lock icon in the ISDE Properties window. This approach simplifies the configuration process and makes it much less error-prone than previous 'manual' approaches. The SSP Configurator Properties tab shows available configuration settings and defaults for all the user-accessible properties. They are also listed in the following tables for easy reference.

Interrupt priorities are one of the properties often requiring change and the settings are available within the Properties window of the associated module. Simply select the indicated module and view the Properties window; the interrupt settings are toward the bottom of the properties list, so scroll down until they become available. Also note the interrupt priorities listed in the Properties window in the ISDE indicates the validity of the setting based on the targeted MCU (CM4 or CM0+). These details are not included in the following tables but are easily visible in the ISDE when configuring interrupt-priority levels.

Note: You may want to open your ISDE, create the module, and explore the property settings in parallel with reviewing the following configuration table settings. This helps to orient you and can be a useful 'hands-on' approach to learning the ins and outs of developing with SSP.

For all tables in this section:

- Rows in yellow highlight must be changed either to enable NAT to work or to match your environment
- Rows in green highlight may be changed and the recommended value and default value are shown.

Table 4. Configuration Settings for the NetX Duo NAT Module

ISDE Property	Value	Description
Minimum count for translation entry	3	Minimum count for translation entry selection
Timeout for translation entry (ticks)	240	Timeout for translation entry selection
Minimum assigned port number for outbound TCP packets	20000	Minimum assigned port number for outbound TCP packets selection
Maximum assigned port number for outbound TCP packets	30000	Maximum assigned port number for outbound TCP packets selection
Minimum assigned port number for outbound UDP packets	20000	Minimum assigned port number for outbound UDP packets selection
Maximum assigned port number for outbound UDP packets	30000	Maximum assigned port number for outbound UDP packets selection
Minimum ICMP query identifier	20000	Minimum ICMP query identifier selection
Maximum ICMP query identifier	30000	Maximum ICMP query identifier selection
Name	g_nat0	Module name
Cache size (bytes)	1024	Cache size selection
Private IPv4 Address (use commas for separation)	192,2,1,66	Private IPv4 Address selection. Modify this for your network environment.
Private IPv4 Netmask (use commas for separation)	255, 255, 255, 0	Private IPv4 Netmask selection
Global network interface index	0	Global network interface index selection
Name of generated initialization function	nat_init0	Name of generated initialization function selection
Auto Initialization	Enable, Disable Default: Enable	Auto initialization selection

Note: The example values and defaults are for a project using the Synergy S7G2 Family. Other MCUs likely will have the same default values and configuration settings.
In some cases, settings other than the defaults for stack modules can be desirable. For example, it might be useful to select different IP addresses and subnet masks. The configurable properties for the lower-level stack modules are given in the following sections for completeness and as a reference.
Most of the property settings for lower-level modules are fairly intuitive and usually can be determined by inspection of the associated properties window from the SSP configurator.

5.1 Configuration Settings for the NetX Duo NAT Lower-Level Modules

Typically, only a small number of settings must be modified from the default for lower-level modules as indicated via the red text in the thread stack block. Notice that some of the configuration properties must be set to a certain value for proper framework operation and will be locked to prevent user modification. The following table identifies all the settings within the properties section for the module.

Table 5. Configuration Settings for the NetX IP Instance

ISDE Property	Value	Description
Name	g_ip0	Module name
IPv4 Address (use commas for separation)	192,168, 0, 2	IPv4 Address selection
Subnet Mask (use commas for separation)	255,255,255,0	Subnet Mask selection
IPv6 Global Address (use commas for separation)	0x2001, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x1	IPv6 global address selection *NOTE: NAT only applies to IPv4 so this property has no effect on NAT processing.*
IPv6 Link Local Address (use commas for separation, All zeros means use MAC address)	0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0	IPv6 link local address selection. *NOTE: NAT only applies to IPv4 so this property has no effect on NAT processing.*
IP Helper Thread Stack Size (bytes)	2048 (Default is 1024)	IP Helper Thread Stack Size (bytes) selection

ISDE Property	Value	Description
IP Helper Thread Priority	3 (Default is 1)	IP Helper Thread Priority selection. Should be lower than the IP instance thread priority.
ARP	Enable	ARP selection
ARP Cache Size in Bytes	520	ARP Cache Size in Bytes selection
Reverse ARP	Enable, Disable Default: Disable	Reverse ARP selection
TCP	Enable	TCP selection
UDP	Enable, Disable Default: Enable	UDP selection
ICMP	Enable, Disable Default: Enable	ICMP selection
IGMP	Enable, Disable Default: Enable	IGMP selection
IP fragmentation	Enable, Disable Default: Disable	IP fragmentation selection
Name of generated initialization function	ip_init0	Name of generated initialization function selection
Auto Initialization	Enable, Disable Default: Enable	Auto initialization selection

Note: The example values and defaults are for a project using the Synergy S7G2. Other MCUs probably have the same default values and available configuration settings.

Note: For the following table only those settings that need to be modified for NAT are shown, as the list is too extensive to be shown in its entirety.

Table 6. Configuration Settings for the NetX Duo Source module

ISDE Property	Value	Description
Maximum Physical Interfaces	2 (Default is 1)	Number of network interfaces enabled
NAT	Enable, Default: Disable	Enable NAT on the IP instance

Note: The example values and defaults are for a project using the Synergy S7G2. Other MCUs probably have the same default values and available configuration settings.

Table 7. Configuration Settings for the NetX Duo Common on nxd

ISDE Property	Value	Description
Name of generated initialization function	nx_common_init0	Name of generated initialization function selection
Auto Initialization	Enable, Disable Default: Enable	Auto initialization selection

Note: The example values and defaults are for a project using the Synergy S7G2. Other MCUs probably have the same default values and available configuration settings.

Table 8. Configuration Settings for the NetX Packet Pool Instance

ISDE Property	Value	Description
Name	g_packet_pool0	Module name
Packet Size in Bytes	1568 (Default 640)	Packet size selection. Setting to 1568 eliminates the requirement for packet chaining for packets larger than 640 bytes
Number of Packets in Pool	16	Number of packets in pool selection. In real world systems this number should be much larger.
Name of generated initialization function	packet_pool_init0	Name of generated initialization function selection

ISDE Property	Value	Description
Auto Initialization	Enable, Disable Default: Enable	Auto initialization selection

Note: The example settings and defaults are for a project using the Synergy S7G2. Other MCUs probably have the same default values and available configuration settings.

Table 9. Configuration Settings for the NetX Port ETHER primary interface

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled (Default: BSP)	Enable or disable the parameter checking
Channel 0 Phy Reset Pin	IOPORT_PORT_09_PIN_03	Channel 0 Phy reset pin selection
Channel 0 MAC Address High Bits	0x00002E09	Channel 0 MAC address high bits selection
Channel 0 MAC Address Low Bits	0x0A0076C7	Channel 0 MAC address low bits selection
Channel 1 Phy Reset Pin	IOPORT_PORT_08_PIN_06	Channel 1 Phy reset pin selection
Channel 1 MAC Address High Bits	0x00002E09	Channel 1 MAC address high bits selection
Channel 1 MAC Address Low Bits	0x0A0076C8	Channel 1 MAC address low bits selection
Number of Receive Buffer Descriptors	8	Number of receive buffer descriptors selection
Number of Transmit Buffer Descriptors	32	Number of transmit buffer descriptors selection
Ethernet Interrupt Priority	Priority 0 (highest), Priority 1:2, Priority 3 (CM4: valid, CM0+: lowest- not valid if using ThreadX), Priority 4:14 (CM4: valid, CM0+: invalid), Priority 15 (CM4 lowest - not valid if using ThreadX, CM0+: invalid). Default: Disabled	Ethernet interrupt priority selection. Do not leave this on Disabled.
Name	g_sf_el_nx0 (Default g_sf_el_nx)	Module name
Channel	1	Channel selection
Callback	NULL	Callback selection

Table 10. Configuration Settings for the NetX Port ETHER secondary interface

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled (Default: BSP)	Enable or disable the parameter checking
Channel 0 Phy Reset Pin	IOPORT_PORT_09_PIN_03	Channel 0 Phy reset pin selection
Channel 0 MAC Address High Bits	0x00002E09	Channel 0 MAC address high bits selection
Channel 0 MAC Address Low Bits	0x0A0076C7	Channel 0 MAC address low bits selection
Channel 1 Phy Reset Pin	IOPORT_PORT_08_PIN_06 (same as primary ethernet driver)	Channel 1 Phy reset pin selection
Channel 1 MAC Address High Bits	0x00002E09	Channel 1 MAC address high bits selection
Channel 1 MAC Address Low Bits	0x0A0076C8	Channel 1 MAC address low bits selection
Number of Receive Buffer Descriptors	8	Number of receive buffer descriptors selection
Number of Transmit Buffer Descriptors	32	Number of transmit buffer descriptors selection

ISDE Property	Value	Description
Ethernet Interrupt Priority	Priority 0 (highest), Priority 1:2, Priority 3 (CM4: valid, CM0+: lowest- not valid if using ThreadX), Priority 4:14 (CM4: valid, CM0+: invalid), Priority 15 (CM4 lowest - not valid if using ThreadX, CM0+: invalid). Default: Disabled *This will be the same value as the primary network driver instance.*	Ethernet interrupt priority selection.
Name	g_sf_el_nx1	Module name
Channel	0	Channel selection
Callback	NULL	Callback selection

Note: The example values and defaults are for a project using the Synergy S7G2. Other MCUs may have different default values and available configuration settings.

5.2 NetX Duo NAT Module Clock Configuration

The ETHERC peripheral module uses PCLKA as its clock source. The PCLKA frequency is set by using the SSP configurator clock tab prior to a build, or by using the CGC Interface at run-time.

5.3 NetX Duo NAT Module Pin Configuration

The ETHERC peripheral module uses pins on the MCU to communicate to external devices. I/O pins must be selected and configured as required by the external device. The following table lists how to select pins within the SSP configuration window with the subsequent table showing an I^2C pin selection example.

Note: The operation mode selection determines the available peripheral signals and the required MCU pins.

Table 11. Pin Selection for the ETHERC Module

Resource	ISDE Tab	Pin selection sequence
ETHERC	Pins	Select Peripherals > Connectivity:ETHERC > ETHERC1.RMII

Note: The selection sequence assumes ETHERC1 is the desired hardware target for the driver.

Table 12. Pin Configuration Settings for the ETHERC1

Property	Value	Description
Operation Mode	Disabled, Custom, RMII (Default: Disabled)	Select RMII as the Operation Mode for ETHERC1
Pin Group Selection	Mixed, _A only (Default: _A only)	Pin group selection
REF50CK	P701	REF50CK Pin
TXD0	P700	TXD0 Pin
TXD1	P406	TXD1 Pin
TXD_EN	P405	TXD_EN Pin
RXD0	P702	RXD0 Pin
RXD1	P703	RXD1 Pin
RX_ER	P704	RX_ER Pin
CRS_DV	P705	CRS_DV Pin
MDC	P403	MDC Pin
MDIO	P404	MDIO Pin

Note: The example values are for a project using the Synergy S7G2 MCU and the DK-S7G2 Kit. Other Synergy MCUs and other Synergy Kits may have different available pin configuration settings.

6. Using the NetX Duo NAT Module in an Application

The following example assumes a system that is already establish with a working IP, ARP, ICMP, TCP and UDP enabled and with the link up. The typical steps in using the NetX NAT Module in an application are:

1. Wait for the IP instance to initialize the driver and have a valid IP address using the nx_ip_status_check API for the primary (“global”) interface.
2. Wait for the IP secondary (“local”) interface to also have a valid IP address at this point using the nx_ip_interface_status_check API.
3. Set the cache full callback to be notified when the NAT translation table it full by calling the nx_nat_cache_notify_set API [Optional].
4. Start packet forwarding by the IP layer in NetX as per NAT protocol by calling the nx_nat_enable API. The IP thread task and NAT services handle the rest.
5. Add rules for inbound traffic e.g. static entries in the cache table, as needed by calling the nx_nat_inbound_entry_create. This can be done before or after enabling NAT services. [Optional]
6. To stop packet forwarding, suspend NAT by calling the nx_nat_disable API.
7. Delete NAT by calling the nx_nat_delete API.

The following figure shows steps in a typical NAT module operational flow.

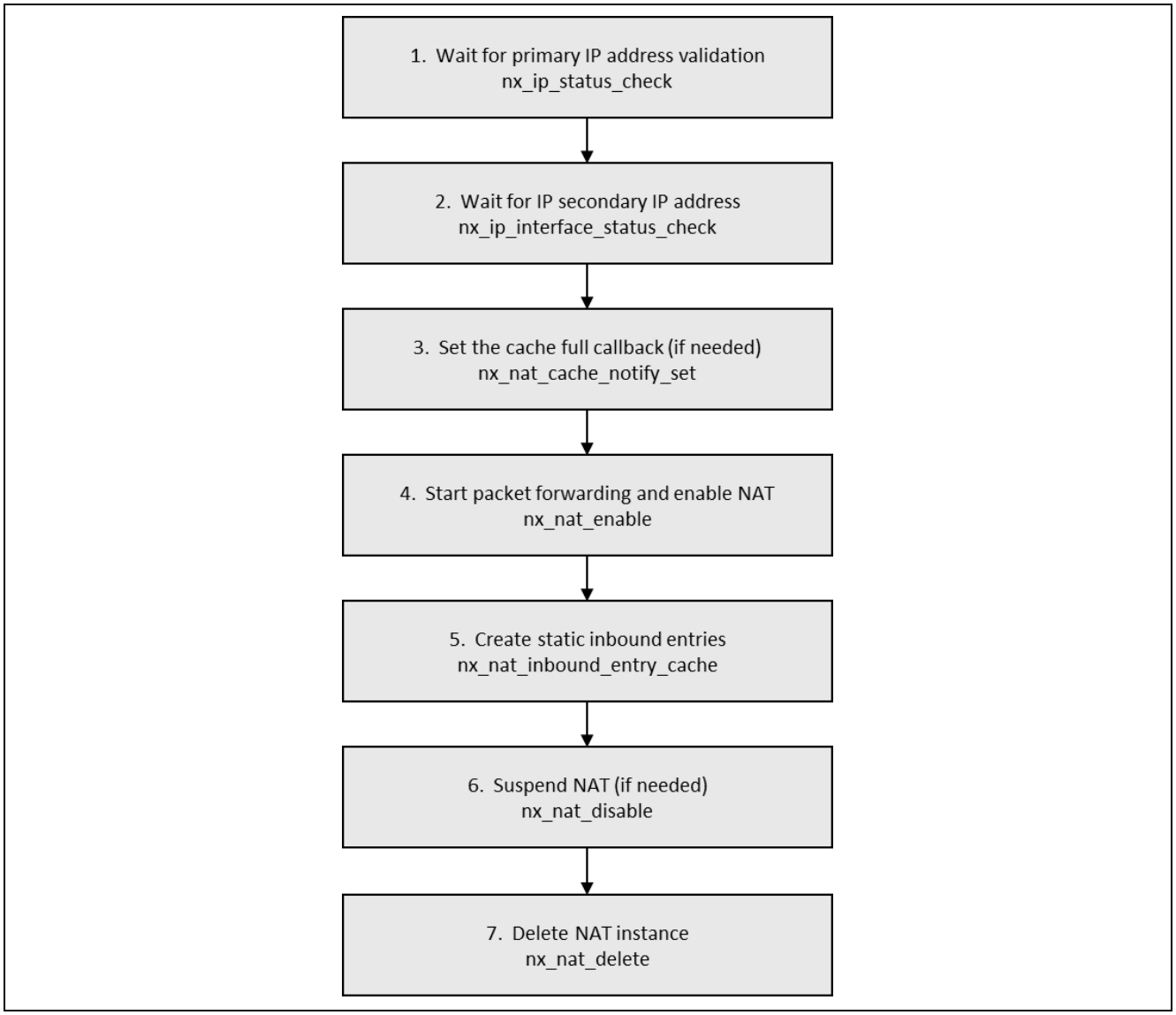


Figure 6. NetX Duo NAT Module

7. The NetX Duo NAT Module Application Project

The application project associated with this module guide demonstrates the steps in a full design. The project can be found using the link provided in the References section. You may want to import and open the application project in the ISDE and view the NetX Duo NAT Module configuration settings. You can also read the code in NXD_NAT_EL_MG_AP used to show the NetX Duo NAT Module APIs in a complete design.

The application project demonstrates typical uses for the NetX Duo NAT Module APIs. The application project main thread entry verifies that both primary (global) and secondary (local) network interfaces are link-enabled, starts NAT processing, and verifies that NetX Duo NAT properly handles inbound and outbound packet transmission between hosts on the global and local networks. The debug output is enabled in this project, so after NAT processing is terminated, the thread entry function prints out the number of errors, if any, that occurred in the NAT router when using the Virtual Debug Console of e^2 Studio and the IAR EW for Synergy Terminal I/O view.

Note: It is assumed you are familiar with using printf() with the Debug Console in the Synergy Software Package. If you are unfamiliar with printf(), see *How do I Use Printf() with the Debug Console in the Synergy Software Package* Knowledge Base article, listed in the References section. Alternatively, the user can see results via the watch variables in the debug mode.

Table 13. Software and Hardware Resources Used by the Application Project

Resource	Revision	Description
e^2 studio	7.3.0	Integrated Solution Development Environment
SSP	1.6.0	Synergy Software Platform
IAR EW for Renesas Synergy	8.23.3	IAR Embedded Workbench® for Renesas Synergy™
SSC	7.3.0	Synergy Standalone Configurator
DK-S7G2	v3.0 to v4.1	Development Kit

Table 14. DK-S7G2 v3.1 DIP Switch Settings

	DIP Switch Settings				
	S5 – Main Board			S101 – Breakout Board	
1	DRAM	OFF		RS	OFF
2	QSPI	OFF		CAN	OFF
3	ENET1	ON		ENET0	ON
4	PMOD	OFF		SD	OFF
5	PBs	OFF		MMC	OFF
6	JTAG	ON		PMDOB	OFF
7	EXP	ON		BLE	OFF
8	BOOT	OFF		CAM	OFF

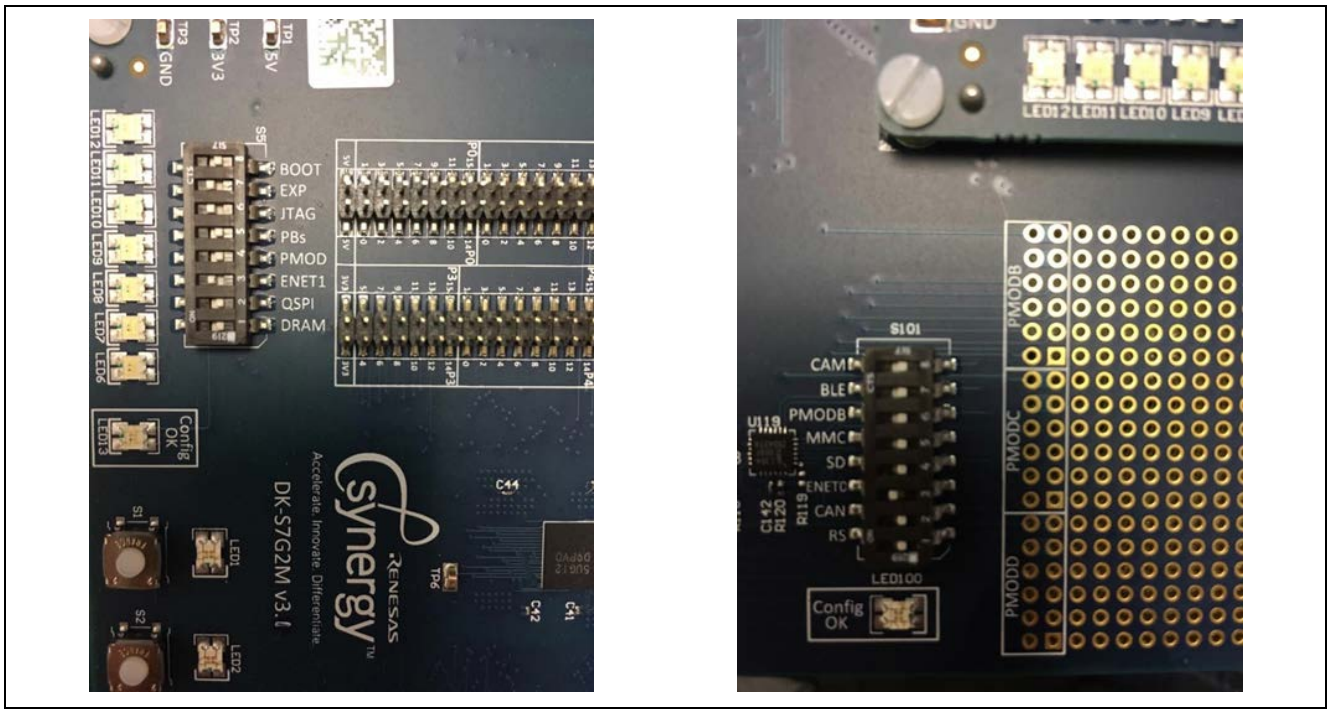


Figure 7. Dip switches on the main and breakout boards of DK-S7G2

Table 15. DK-S7G2 v4.1 DIP Switch Settings

	S6	
1	BOOT	OFF
2	ETHERNET0	ON
3	ETHERNET1	ON
4	e.MMC	OFF

S7	
PMOD A	OFF
JTAG	ON
RS-XXX	OFF
CAN	OFF

S8	
PMODB	OFF
CAMERA	OFF
SD SOCKET	OFF
LCD	OFF

S9	
QSPI FLASH	OFF
SDRAM	OFF
USR BTNS	OFF
EXTERNAL LCD	OFF

The following figure shows a simple flow of the application project.

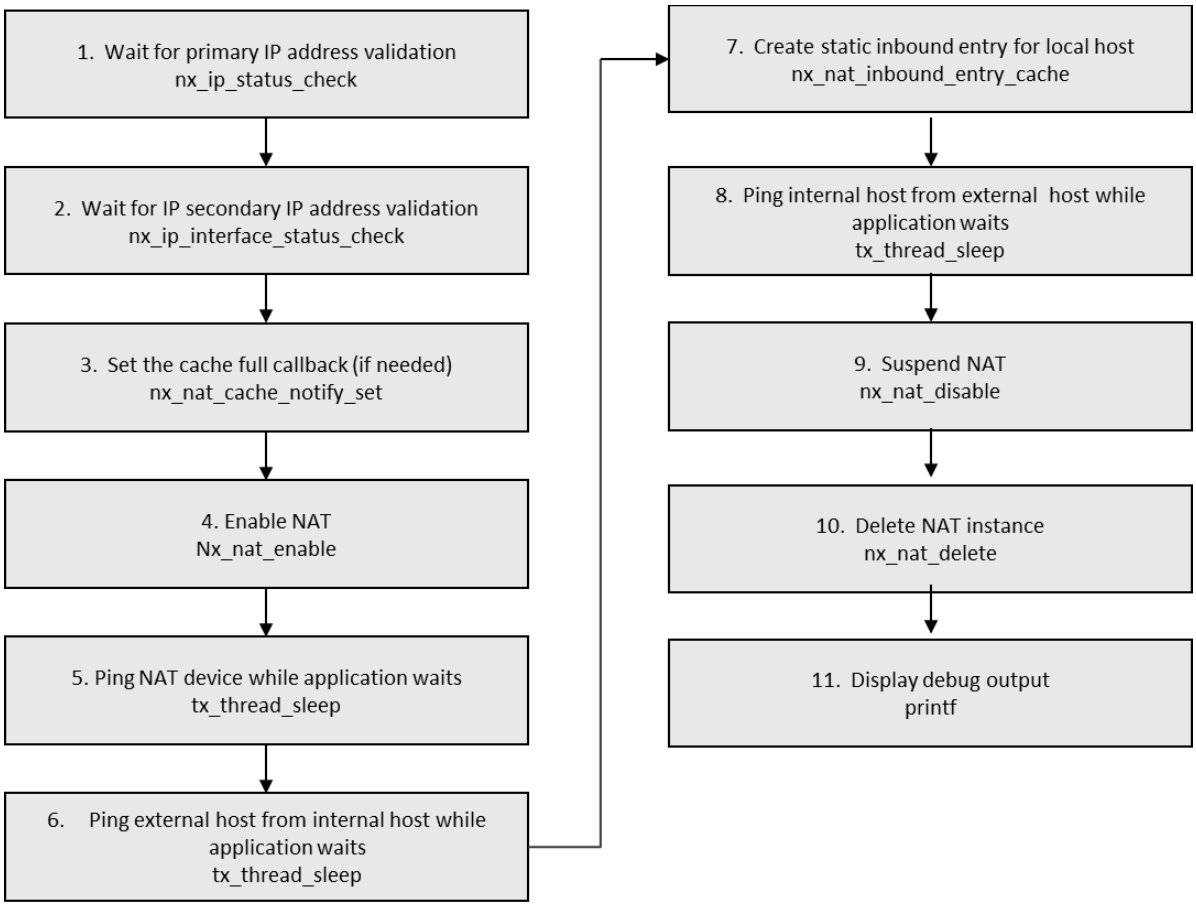


Figure 8. Application project operation flow

After the `nat_thread0_entry` function verifies the global and local network interfaces are enabled, it calls the `nat_process` function in the `nat_netxduo_app_mg_ap.c` file.

This `nat_process` function enables the previously created NAT instance (from Synergy auto-generated code). In three successive trials, the function waits for the host on one interface to ping a host on the other interface, depending if:

1. Packet is outbound (from local network to global), *or*
2. Packet is inbound (global to local) *and*
3. The NAT has previously created an inbound rule for the host on the local network.

The NAT router ‘consumes’ the packet. That is, the NAT router applies a translation to the source address of the packet and transmits it to the destination host on the other network. If the NAT router does not consume the packet, it lets NetX Duo process the packet.

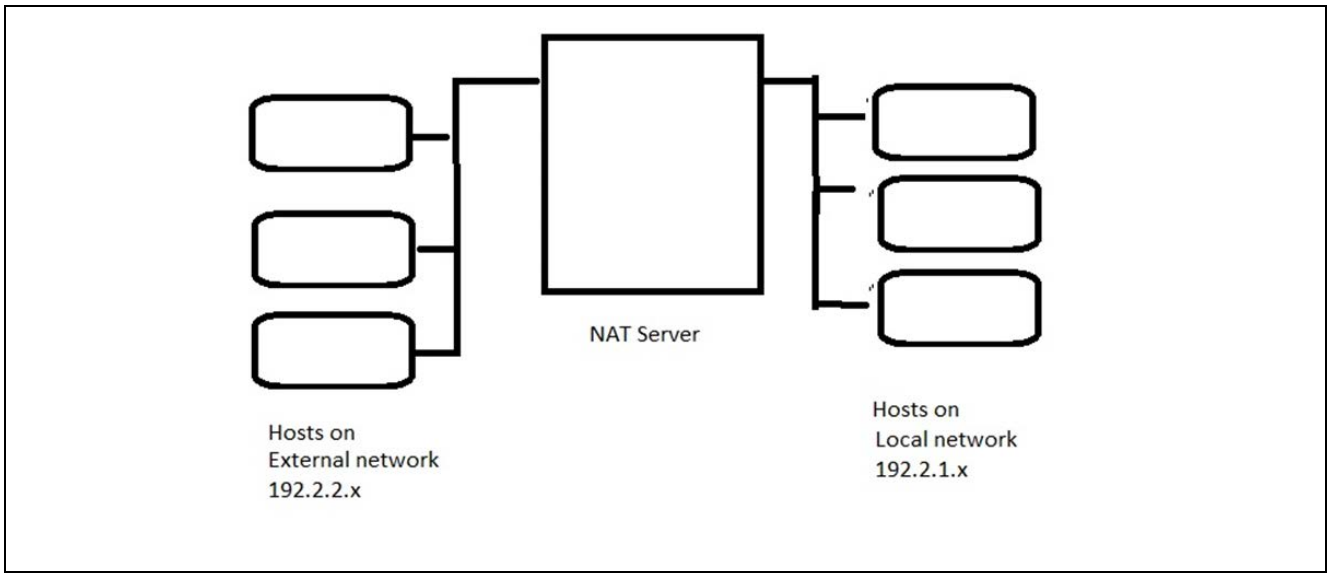


Figure 9. Topography of the Project

The topography shows multiple hosts on both the internal and external network, but the NetX Duo NAT module guide project uses only one host on each network.

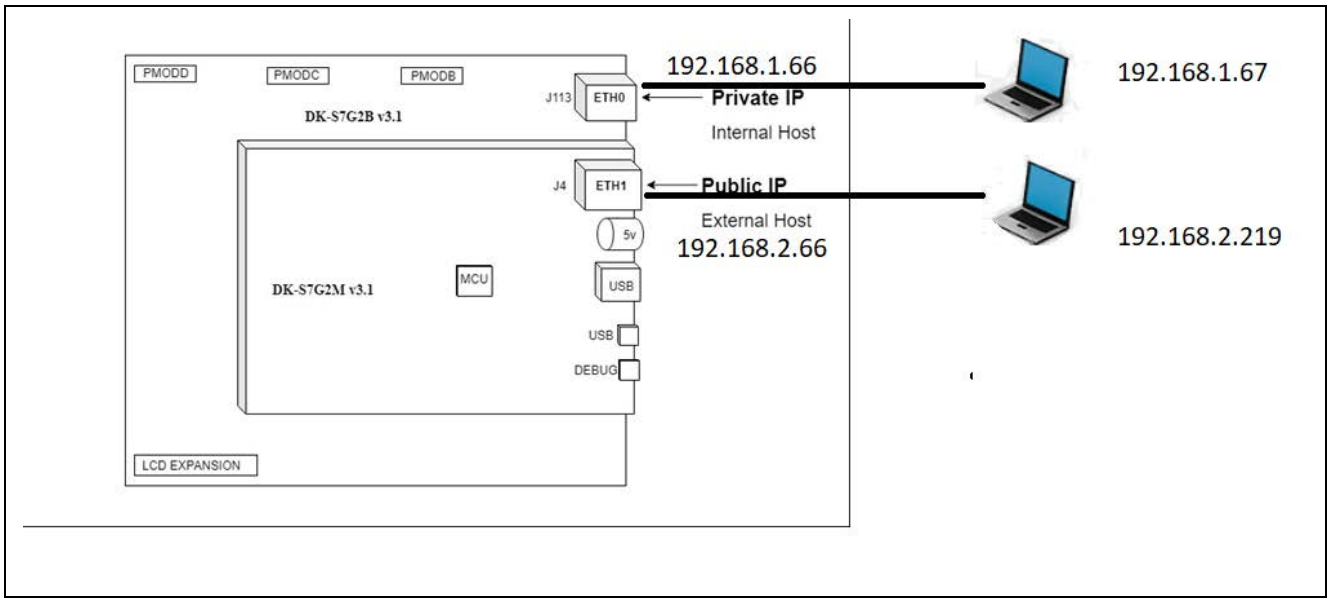


Figure 10. Host PCs connection on DK-S7G2 v3.1

7.1 NAT forwarding activity

The NAT router is connected to the external ‘public’ network via the Ethernet port on the J4 connector. The server is connected to the internal ‘local’ host on the Ethernet port on the J113 connector.

For DK-S7G2 v4.1, the external network is connected via the Ethernet 1 port on the J17 connector. The server is connected to the internal host on the Ethernet 0 port on the J16 connector.

To successfully run the example project, you need two hosts (one on each network) that are able to execute a ‘ping’ command, such as within a terminal window (for example, two PCs).

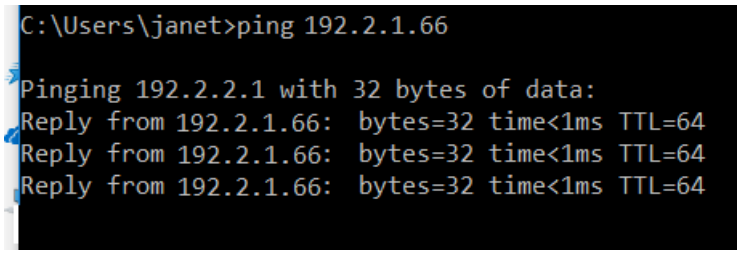
Carefully check that the hosts you are using for these tests are located within the default test networks. These hosts are indicated in the configuration for the NAT module (for the ‘local’ network) and the IP instance module (for the ‘private’ network).

This project sets 192.2.1.66 and 192.2.2.66 as the IP address for the external and internal interface of the NAT router, respectively. For whatever values for IP addresses you choose, make your test hosts do not have the same IP addresses. They should have the same network addresses; for example, 192.2.1.x and 192.2.2.x.

Be careful to set the ping command to an n of 1 (-n 1). The default is to send three pings. There is no direct harm in doing that —but, the counters in the application project are assuming a single ping for each test.

Test #1:

Ping the NAT router from both PCs to verify connectivity and firewall issues are not a problem. For all these tests you can verify that the ping request succeeded by the output in the command shell.



Test #2:

The application waits to while you ping the external host from the internal host. In this project that host has IP address 192.2.2.219 (will vary on your system). So, in a command shell on the internal host, type this command:

ping 192.2.2.219

Internal details:

The outbound ping packet's source IP address is changed to the NAT router's IP address on the global network (primary interface) before being forwarded, and an entry is automatically added into the NAT table for this translation. When the echo reply is received by the NAT router using the NAT router's global IP address, the NAT router checks the NAT translation table (or cache), finds the matching entry, and changes the destination address to the local host IP address.

Test #3

The application first creates a rule for NAT to forward ICMP packets received on the external network to local host, which in this project has an IP address of 192.2.1.67 (will vary in your project). Conceptually, that looks like this:

Table entry*

Direction	protocol	local host IP	translated host IP
Inbound	ICMP	192.2.1.67	192.2.2.66

Ping the local host at 192.2.1.67 from the external host on the NAT router's global network (192.2.2.219**)** by actually pinging the NAT router address, which in this project 192.2.2.66; not the actual local host IP address as the destination IP address. This works because NAT has a rule for mapping this IP address. Otherwise, the external PC does not know the local IP address or have direct access to the local network.

ping 192.2.2.66

When the application detects the ping request, NAT checks the translation table for a rule matching the external IP address with the local IP address, and then forwards the packet; changing the destination to 192.2.1.67, but keeping the source IP address (192.2.2.219) the same.

This module guide project uses ICMP packets for simplicity. No source or destination ports are involved because the ICMP protocol does not include ports. For ICMP, which does not use port numbers, port numbers are set to zero in the `nx_nat_inbound_entry_create` API. For UDP and TCP translation; however, the local source port and the translation port are also included in the NAT translation and in creating the inbound rule. It is kept as a suggested exercise in chapter 8, Customizing the NetX Duo NAT Module project.

8. Customizing the NetX Duo NAT Module for a Target Application

To customize NetX Duo NAT to your project you can set up TCP and UDP socket applications. For server applications, you can create inbound rules for TCP and UDP servers to accept packets on fixed ports, where clients (external hosts) initiate a connection. For the choice of ports when creating inbound rules, NAT automatically verifies the supplied ports in the `nx_nat_inbound_entry_create` are available.

You can also add more logic to the cache full notify project, rather than just increment the error counter as the example does in the module guide project. For instance, it can scan the table and delete older entries, rather than just increment the error count as it does in the project.

9. Running the NetX Duo NAT Module Application Project

To run the NetX Duo NAT Module application project and execute it on a target kit, simply import it into your ISDE, compile, and run debug. See the enclosed *Renesas Synergy™ Project Import Guide* (r11an0023eu0121-synergy-ssp-import-guide.pdf), to import projects into e^2 studio or IAR EW for Renesas Synergy, along with building/running the application project.

To implement the NetX Duo NAT Module application in your own project, follow the steps for defining, configuring, auto-generating files, adding code, compiling, and debugging the project on the target kit. Following these steps is a hands-on approach that can help make the development process with SSP more practical.

Note: The following steps are described in sufficient detail for someone experienced with the basic flow through the Synergy development process. If these steps are unfamiliar, see the *SSP User's Manual* for instructions on how to accomplish these steps.

To create and run a NetX Duo NAT Application Project, follow these steps:

1. Create a new Renesas Synergy project for the DK-S7G2 called **NetX_Duo_NAT_AP**.
2. Select the **Threads** tab.
3. Click on the **+** icon in the threads panel.
4. Set the stack size to **2048** and set the thread instance to **my_nat_thread0**.
5. In the thread stack pane, click the + icon and choose **X-Ware** -> **NetX Duo** -> **Protocols** -> **NetX Duo NAT**. Also add the NetX Duo Source component by clicking **Add NetX Duo Source** box.
6. Add a driver instance for the primary interface, click **Add NetX Duo** driver and choose **NetX Port Ether**.
7. Add an additional driver instance for the secondary interface by clicking anywhere in the configurator Pane and then clicking on the (+) icon in the upper right -> Framework -> Networking -> NetX Port Ether.

Note: In later releases, there may be an Add NetX Duo Network Driver for Private Network automatically attached the NAT instance in the configurator pane. If so, click -> New -> NetX Port Ether. Do **not** choose Use (the existing NetX Port Ether). NAT will not work with one driver instance!

8. Set the table properties for the NAT, IP, packet pool, both driver instances, and the NetX Duo Source. For instructions, see chapter 5, Configuring the NetX Duo NAT Module.
9. Set the DIP switches on DK-S7G2 board per chapter 5, Configuring the NetX Duo NAT Module.
10. Set the pin configuration or copy the **S7G2-DK.pincfg** file from the module guide project folder to your project. For instructions, see chapter 5, Configuring the NetX Duo NAT Module.
11. Click the **Generate Project Content** button.
12. Edit the thread entry function created automatically by the Generate Project Content.
    To use the source code from the module guide project in your project, follow these steps:
    A. Add the two files to the project\src folder in the windows file explorer: `nat_netxduo_app_mg_ap.c` and `nat_netxduo_app_mg_ap.h`.
       The files should appear automatically in e^2 studio.
    B. In the thread entry function for your thread instance, copy the code from `nat_thread0_entry` into your thread entry function, but do not overwrite the #include for your thread instance header file; it is

auto-generated by Synergy. You write your own code to check that both interfaces are initialized, and the link is enabled.

C. Your thread entry function should call the `nat_process` function to start NAT processing. When the nat_process returns, it disables and deletes the NAT instance and checks the error count returned from nat_process, as done in `nat_thread0_entry.c`.

D. If you want the debug output using the semi-hosting utility, #define SEMI_HOSTING, either at the top of your thread entry function or in the list of preprocessors defined for the project. For e² studio, use the conditional compilation used in `nat_thread0_entry.c` to initialize the semi-hosting utility:

```
#ifdef SEMI_HOSTING
#ifdef __GNUC__
    if (CoreDebug->DHCSR & CoreDebug_DHCSR_C_DEBUGEN_Msk)
    {

        initialise_monitor_handles();
    }
#endif
#endif

And to print debug output put these conditions around your printf statement(s) in
either IAR or e2 studio:
#ifdef SEMI_HOSTING
    if (CoreDebug->DHCSR & CoreDebug_DHCSR_C_DEBUGEN_Msk)
    {
        if ((error_counter != 0) || (nat_errors != 0))
        {

            printf("Set up error(s) %d; errors with NAT enabled %d.\n",
                    error_counter, nat_errors);
        }
        else
        {
            printf("NAT router application completed with no errors.\n");
        }
    }
#endif
```

13. To step through code, modify the optimization to **None -O0** (the default to this value is -O2 for Optimize more):

    To do so, right click the project and choose Properties -> **C/C++ Build** -> **Settings** -> **Optimization**. Set the Optimization Level to **None (-O0)**.

    For better performance and reduced code size, once debugging is complete, reset this to -O2.

14. Right-click the project and choose **Build Project**.
15. Connect the hosts to the two Ethernet adapters on the board. (See The NetX Duo NAT Module Application Project for instructions). Connect a micro-USB cable to J17 on DK-S7G2 to download and run the application.
16. Right-click the project and choose Debug as -> **Renesas GDB Hardware Debugging**.
17. Run the application. Use the following steps to verify NAT forwarding packets.

```
/* Test #1: Ping the NAT router.
   Test #2: Ping the external host from the local host.
   Test #3: Ping the local host from the external host.*/
```

LOCAL PC	NAT router	EXTERNAL (GLOBAL) PC
192.2.1.67	192.2.1.66 192.2.2.66	192.2.2.219

Test 1: Send Echo Request locally

Local PC:

Ping from Local host (PC) 192.2.1.67 to NAT router (192.2.1.66)

Send ping to NAT router → Receive packet on local interface 192.2.1.66 target

Receive response ← Respond directly back to local PC 192.2.1.67 target

External PC:
Ping from External host (PC) 192.2.2.219 to NAT router (192.2.2.66)

Send ping to NAT router → Receive packet on external interface 192.2.2.66 target

Receive response ← Respond directly back to external PC 192.2.2.219 target

Test 2: Ping external host from local host
Local PC
Ping from Local host (PC) 192.2.1.67 to external host (192.2.2.219)

Send ping to 192.2.2.219 → NAT forwards to → External host receives packet from 192.2.1.67

Receive response from 192.2.2.219 ← NAT forwards to internal host ← External sends response to 192.2.1.67

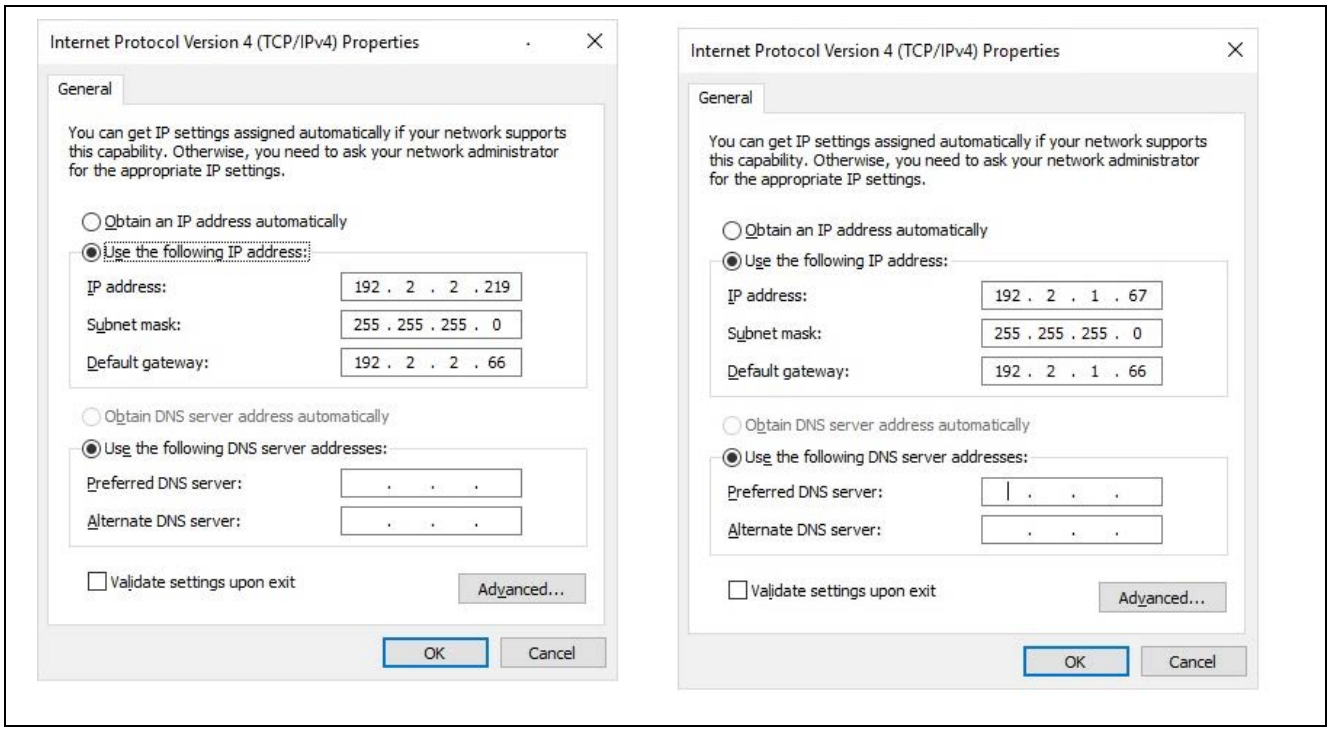
Test 3: Ping local host from external host
External PC
Ping from 192.2.2.219 to 192.2.1.66

Send ping to 192.2.2.66 → NAT forwards to → Internal host receives packet from 192.2.2.219 via 192.2.2.66

Receives response from 192.2.2.66 ← NAT forwards to external host (first maps to 192.2.1.67) ← External host sends response to 192.2.2.66

Note: Test #3 would make more sense, functionally, if the local host was, for example, a TCP server listening for packets on port 80. When the external host initiates a request (sends a packet to it), it would send the packet to the NAT router (192.2.2.66) on port 80. The NAT router would identify the combination of the IP address and port as belonging to the local host 192.2.1.67 and forward the packet to it using the 192.2.1.67 IP address.

The two hosts exchanging packets across NAT may need to set their default gateway to the IP interface address of NAT. So, if the external IP address (primary interface) is 192.2.2.66 that would be the default gateway for the external host, and if the internal IP address (secondary interface) is 192.2.1.66 that would be the default gateway for the internal host.



The default gateway settings are for external and internal interfaces. You can make it on the two PC's connected as part of the set up.

- Set External PC 192.2.2.219 / 255.255.255.0 /GW: 192.2.2.66.
- Set Internal PC 192.2.1.67 / 255.255.255.0 /GW: 192.2.1.66

18. Refer comments section in nat_netxduo_app_mg_ap.c and nat_thread0_entry.c, to debug through the application.
    a. In nat_netxduo_app_mg_ap.c for test cases 1,2,3 in the "**process_nat"** function change status from 0xffff to 0x0 and test one by one either during running or by modifying and compiling it for each test cases separately.
    b. After the three tests are done, make changes to nat_thread0_entry.c make spin = NX_FALSE during running, or by modifying and compiling it to get the end results on the console.
19. You may also need to modify the firewall settings on these hosts to accept ping requests. That exercise is beyond the scope of this document and depends entirely on the specific PC and operating system environment. There is copious information available on the Internet.
20. Press **F8** or the green arrow in e^2 studio, or **F5** in IAR EW for Renesas Synergy to resume the program on to each of these breakpoints.

The output, if there are no errors, can be viewed in the Renesas Debug Console as follows:

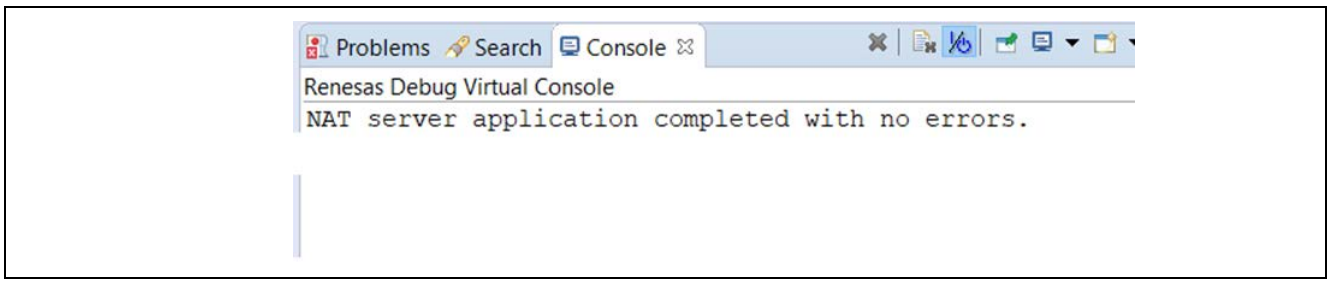


Figure 11. Example Output from NetX Duo NAT Application Project

10. NetX Duo NAT Module Conclusion

This module guide has provided all the background information needed to select, add, configure, and use the NAT module in an example project. Many of these steps were time consuming and error-prone activities in previous generations of embedded systems. The Renesas Synergy Platform makes these steps less time consuming and removes common errors, like conflicting configuration settings or incorrect selection of lower-level drivers. The use of high-level APIs (as demonstrated in the application client and server projects) illustrates additional development-time savings by allowing work to begin at a high level and avoiding the time required in older development environments to use, or, in some cases, create, lower-level drivers.

11. NetX Duo NAT Module Next Steps

After you have mastered a simple NetX Duo NAT module project you may want to review a more complex example. Other application projects and application notes that demonstrate NetX Duo NAT Module use can be found as described in the References section.

12. NetX Duo NAT Module Reference Information

SSP User Manual: Available in HTML format in the SSP distribution package and as a pdf from the Synergy Gallery.

Links to all the most up-to-date NetX Duo NAT Module reference materials and resources are available on the Synergy Knowledge Base: https://en-support.renesas.com/knowledgeBase/17913008.

Website and Support

Visit the following vanity URLs to learn about key elements of the Synergy Platform, download components and related documentation, and get support.

Synergy Software	www.renesas.com/synergy/software
Synergy Software Package	www.renesas.com/synergy/ssp
Software add-ons	www.renesas.com/synergy/addons
Software glossary	www.renesas.com/synergy/softwareglossary
Development tools	www.renesas.com/synergy/tools
Synergy Hardware	www.renesas.com/synergy/hardware
Microcontrollers	www.renesas.com/synergy/mcus
MCU glossary	www.renesas.com/synergy/mcuglossary
Parametric search	www.renesas.com/synergy/parametric
Kits	www.renesas.com/synergy/kits
Synergy Solutions Gallery	www.renesas.com/synergy/solutionsgallery
Partner projects	www.renesas.com/synergy/partnerprojects
Application projects	www.renesas.com/synergy/applicationprojects
Self-service support resources:	
Documentation	www.renesas.com/synergy/docs
Knowledgebase	www.renesas.com/synergy/knowledgebase
Forums	www.renesas.com/synergy/forum
Training	www.renesas.com/synergy/training
Videos	www.renesas.com/synergy/videos
Chat and web ticket	www.renesas.com/synergy/resourcelibrary

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jan.04.18	—	Initial release
1.01	Jun.04.19	—	Updated for SSP 1.6.0. Added info for DK-S7G2 v4.1.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.